

**UNITED STATES PATENT APPLICATION**

**Title:           SYSTEM, METHOD AND APPARATUS FOR PREPARING A  
TABLE FOR A CELL SCHEDULER**

**Inventor:       Jaroslaw Kogut**

**Filing Date:    December 31, 2003**

**Docket No.:    P16511**

Prepared by: Richard W. James for  
Buckley, Maschoff, Talwalkar & Allison LLC  
Five Elm Street  
New Canaan, CT 06840  
(203) 972-0006

5                   SYSTEM, METHOD AND APPARATUS FOR PREPARING  
                    A TABLE FOR A CELL SCHEDULER

BACKGROUND

Asynchronous Transfer Mode (ATM) includes a protocol for transmitting  
information over a network such as the Internet. ATM currently includes numerous  
10 standards and those standards may be found at <http://www.atmforum.com>. In ATM the  
flow of information, which may be referred to as traffic, may be specified as a number of  
cells transmitted per second. Cells are currently 53 byte sets of data that may be  
transmitted over an ATM link. Moreover, currently five of those 53 cell bytes are utilized  
to carry header information and the remaining 48 cell bytes are used to carry payload  
15 information that is being transmitted.

Modern networks may carry a variety of traffic, some of which may have strict  
timing requirements. With network protocols such as the ATM protocol, data traffic,  
including voice and video traffic, traverses a network via network devices such as ATM  
switches that assure that the appropriate connections are established, that the data  
20 reaches its destination via its designated connections, that bandwidth constraints are  
not exceeded and that timing requirements are met. Traffic flow through these network  
devices is often managed by a cell scheduler. In an ATM device such as an ATM  
switch, schedulers are used to manage the timing of cell transmission. Traffic with rigid  
timing requirements, such as real-time voice and video traffic must be accommodated in  
25 those schedulers along with other data traffic. The ATM protocol addresses those  
timing issues by designating special classes or categories of service for handling time  
sensitive data. ATM traffic thus generally has a quality of service (Qos) guarantee that  
varies depending on the specific requirements of the traffic type. Two classes of service  
that maintain end-to-end timing relationships across an ATM network are the constant  
30 bit rate (CBR) class of service and the variable bit rate real-time (VBR-rt) class of  
service, which are described in the ATM standards.

5           ATM cell schedulers generally include tables having slots that are populated according to the requirements of the traffic being scheduled such as CBR and VBR-rt along with other types of traffic having less demanding timing requirements. Thus, each cell to be transmitted may be identified in a slot of the table and the identified cells may be transmitted in the order in which they are identified in the table.

10           ATM traffic can enter and exit a network device such as a switch via one or more ports, with each port carrying a sub-stream of traffic. Certain of those sub-streams may furthermore be constant-bandwidth sub-streams and carry real-time traffic. Constant bandwidth sub-streams also adhere to timing requirements to assure constant bandwidth. To maintain proper timing in a scheduler table that sequentially runs  
15 through all slots in the table to find the next ATM cell to transmit, the distance between all consecutive table entries of a sub-stream of such real-time data should be equal. That distance between entries related to a sub-stream may be referred to as a step, may be measured in terms of a number of slots between entries for that sub-stream in the table, with each consecutive entry for a particular sub-stream of real-time data being  
20 a step apart, and may be referred to as Z.

When populating such a table with information from numerous sub-streams in accordance with timing requirements, the proper slots must be available and identified for population. Checking for slot availability generally becomes more laborious as more entries are made in the table and the table becomes more densely populated.

## 25                           BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, wherein like reference numerals are employed to designate like components, are included to provide a further understanding of the preparation of a table for a cell scheduler, are incorporated in and constitute a part of this specification, and illustrate embodiments of the preparation of a table for a cell  
30 scheduler that together with the description serve to explain the principles thereof.

In the drawings:

5        Figure 1 illustrates an example of a swap operation on the bits of a binary representation;

Figure 2 illustrates an embodiment of a method of performing a swap operation to configure a data structure being populated with at least one sub-stream of constant bandwidth;

10      Figure 3 illustrates an embodiment of a port shaping table;

Figure 4 illustrates an embodiment of a swap operation as used in a network device to populate a table in a scheduler; and

Figure 5 illustrates an embodiment of a network in which a swap operation may be employed.

15 DETAILED DESCRIPTION

Systems, apparatuses, and methods for preparing a table for a cell scheduler are provided. Those cells may carry information and that information may, furthermore, comprise any data capable of being represented as a signal, such as an electrical signal, optical signal, acoustical signal and so forth. Examples of information in this context may include voice communications, images, video, text, data and so forth, with real-time voice communications and video and audio transmitted during, for example, a multi-media conference being examples of information carried in real-time sub-streams. Details, features, and advantages of the preparation of a table for a cell scheduler by employing a swap will become further apparent in the following detailed description of embodiments thereof.

Any reference in the specification to “one embodiment,” “a certain embodiment,” or a similar reference to an embodiment is intended to indicate that a particular feature, structure or characteristic described in connection with the embodiment is included in at least one embodiment of the preparation of a table for a cell scheduler. The appearances of such terms in various places in the specification are not necessarily all

5 referring to the same embodiment. References to “or” are furthermore intended as inclusive so “or” may indicate one or another of the ored terms or more than one ored term.

10 In a certain embodiment of a scheduler table, during configuration of the table by populating the table with entries, for each potential entry in the scheduler table for the sub-stream, a check is performed to determine whether there is an unused slot in the table at the appropriate step from the last entry for that sub-stream. The slot in the table may alternately be referred to as a cell transmission slot. If there is a slot conflict because another cell is indicated at that slot, the slot may be incremented and another search performed for an available slot at that location. As the table becomes more  
15 populated with entries, that slot selection process may become require a great deal of processing time and effort to find an appropriate open slot. Populating a table through use of a swap operation may eliminate the iterative search of that embodiment.

Figure 1 illustrates a swap operation 100 as used to index an entry for a table. In a scheduler for a network device that carries real-time data, one or more data structures  
20 such as tables may be configured such that entries into the tables corresponding to the real-time data are read according to precise timing parameters that assure proper delivery of the data. For example, cells may be identified in slots of the table and those cells may be transmitted from the network device to a next appropriate device on the network in the order the cells are identified in the table. Populating those tables can be  
25 performed quickly and efficiently by applying a swap operation that recognizes a relationship between the entries already inserted in the table and the next available slot in the table. The swap operation relies on the swapping of the values in the lower portion of a binary representation of the number of entries already in a table being populated to arrive at an index for a new entry into the table. Configuring a table through  
30 application of the swap operation assures that proper spacing relationships are maintained to meet timing requirements of constant bandwidth traffic, and that all available slots in a table are filled, without having to worry about slot conflict, provided the number of entries of an additional sub-stream to be entered will not exceed the

5 remaining entry capacity of the table. Time and effort is saved by using the swap operation because, by its use, it becomes unnecessary to verify that slots are available each time a new sub-stream is introduced. For example, ATM schedulers that are designed in conformance with, for example, the ATM Forum specification for traffic management TM4.1 may thus use this swap operation in populating scheduler tables.

10 The ATM Forum specification for traffic management TM4.1 may be found in the ATM standards.

The swap operation 100 is applied to a population of data structures and in particular to cell scheduling tables in schedulers for ATM cells in the example provided herein. Those of ordinary skill in the art will appreciate, however, that the swap

15 operation described in connection with scheduler table population may be applicable to other systems, methods or apparatus having timing, spacing, or entry requirements comparable to those of scheduler tables.

In Figure 1, 13 bits are available for use to express a binary number L representing the number of entries actually residing in a table having N total slots available for entries. N may be expressible as a power of two where  $N = 2^x$

20 where x is the power. The swap operation 100 illustrates the swapping of x lower bits of a binary representation of L. In the swap operation 100 illustrated in Figure 1, the total slots for entries in the table, N, is 2048 such that the exponent value x therefore equals 11 ( $2^{11} = 2048$ ). The value of x (11 in Figure 1) thus provides the number of lower bits of the

25 binary representation of L 106 that are to be swapped. Accordingly, Figure 1 shows the lower 11 bits 102 of binary L, which represents the number of entries already in the table, being swapped. Additional bits 104 are not swapped in this example.

In the swap operation 100 illustrated in Figure 1, counting is performed with zero indicating a zero value and 1 indicating a non-zero value. The x lower bits of the binary

30 representations of L, here the 11 lower bits, are the bit positions for 2 to the 0 power through 2 to the 10th power respectively from the rightmost bit. L, in the example illustrated in Figure 1, is therefore equal to eight as expressed in a binary representation of L 106.

5           To perform the swap, the  $x$  bits of the binary representation of  $L$  are exchanged around an axis located at a center at a middle bit of the  $x$  bits of the binary representation of  $L$  when  $x$  is an odd number or at a center between two middle bits of the  $x$  bits of the binary representation of  $L$  when  $x$  is an even number. Thus, where  $x$  is an odd number, the middle bit is a bit of the  $x$  bits wherein there is an equal number of  
10 the remaining  $x$  bits on either side of the middle bit. Where  $x$  is an even number the center between two middle bits is a position wherein an equal number of the  $x$  bits lie on either side.

          The center bit is thus a bit occupying a position in the binary representation of  $L$  at an  $x$  divided by two power where an odd number of bits are included in  $x$  and the two  
15 center bits are a bit occupying a position in the binary representation of  $L$  at an  $x$  divided by two rounded down power and a bit occupying a position in the binary representation of  $L$  at an  $x$  divided by two rounded up power where an even number of bits are included in  $x$ .

          Accordingly, in the example of Figure 1, the center bit is bit six and there are five  
20 bits on each side of bit six. Thus, the bit value in the zero power position, the first bit position, is swapped with the bit value in the tenth power position, the eleventh bit position. The bit value in the first power position, the second bit position, is swapped with the bit value in the ninth power position, the tenth bit position. The bit value in the third position is swapped with the bit value in the ninth position, the bit value in the  
25 fourth position is swapped with the bit value in the eighth position, the bit value in the fifth position is swapped with the bit value in the seventh position, and the bit value in the sixth position is left unchanged because the sixth position lies on the axis.

          In the example illustrated in Figure 1, the fourth bit position is the only position of the  $x$  bit positions being swapped that has a value of one. Thus, in a post-swap  
30 arrangement of  $L$ , referred to herein as  $L'$  108, the only bit position of the  $x$ , or 11, bits being swapped that has a value of one is the bit in the eighth position, which is the seventh power position. After the swap operation, the resultant binary value of  $L'$  is 128.

5           That  $L'$  value may then be used as an index for a new entry into the table. For a sub-stream with a constant bandwidth, the step  $Z$ , i.e., the interval between consecutive slots for consecutive components of the same sub-stream in the table may then be calculated to obtain the indices for the remaining components of the sub-stream to schedule that sub-stream. A successive integer multiple of the step, starting with a multiple of one, may be added to the first index value to locate a slot for each successive component, respectively.

          The step  $Z$  may be calculated by dividing the total number of slots in the table  $N$  by the number of components,  $K$ , of the sub-stream to be entered. For example, using the values in Figure 1 to populate a scheduler table such as those used in an ATM device such as a switch, adding a sub-stream with four components,  $K$ , to a table with eight existing entries,  $L$ , would result in a first index value of  $L' = 128$ . The step  $Z$  would be  $N$  divided by  $K$  or  $2048 / 4 = 512$ , so the index for the second component would be  $L' + Z$  or 640, the index for the third component would be  $L' + 2Z$  or 1152, and the index for the fourth component would be  $L' + 3Z$  or 1664. The general formula for deriving the indices of a sub-stream with  $K$  components is thus  $L' + wZ$ , where  $w = \{0, 1, \dots, K-2, K-1\}$ . The value of  $w$  is therefore progressively increased by incrementing its value and that value of  $w$  is multiplied by the step  $Z$  to determine an index of a slot in the table in which each successive component of a sub-stream is to be entered, respectively.

          Figure 2 illustrates an embodiment of a swap operation 200 used to configure a data structure being populated with at least one sub-stream of constant bandwidth such as real-time data. In tables such as those in ATM schedulers, the tables may be populated with data from various sub-streams of constant bandwidth that arrive from various sources. The maximum number of possible entries in a table,  $N$ , is again a value expressible as a power of two where  $N$  equals two to the  $x$  power and  $x$  is a natural number. The components,  $K$ , of a sub-stream are the units of bandwidth that make up the sub-stream.  $K$  is also expressible as a power of two in the swap operation 200, e.g.,  $K$  equals two to the  $y$  power where  $y$  is a natural number.



5           In ATM traffic the units of bandwidth in a sub-stream may, for example, be cells or may correspond to ATM cells. In an ATM scheduler table the sub-stream may furthermore be traffic on a particular port transmitting at a constant bit rate. That traffic may, moreover, include real time traffic such as voice or video traffic.

10           In an embodiment of the swap operation 200, a value is obtained from the swap operation 200 that provides an index for one or more component entries into corresponding slots in a table. The table utilized in this example may be, for example, a configuration table such as a port shaper table that assures timely transmission of ATM cells in a scheduler of an ATM switch or other device.

15           Port shaper tables are generally configured such that ports with constant bandwidth requirements are allocated proper entries in a table. In a port shaper table, the slots may correspond to units of bandwidth associated with sub-streams of traffic, the sub-streams being from particular ports on a network device such as an ATM switch, and the traffic coming from a particular port having a constant bandwidth. When utilizing a port shaper table, the number of components, K, would generally be the  
20           number of units of bandwidth in a sub-stream, and in an ATM sub-stream would generally correspond to the number of cells in the sub-stream. Alternately, K could represent a transmission rate associated with a sub-stream, such as a port with a constant bandwidth and N could represent the maximum number of slots processed per unit of time. For example a configuration table could be used to process 2048 ATM  
25           cells per second (N=2048), and a port could have an associated rate of 64 cells per second.

30           An entry that may be placed into the configuration table could include an identifier used to identify a cell for transmission. The identifier may serve as a pointer to a component of a sub-stream, such as an ATM cell, in another table, such as a cell transmit table. In that way, as a processor loops through a port shaper table, entries that point to a cell for transmission are processed in accordance with timing requirements. Entries into a configuration table could alternately be actual cells, or units of bandwidth that correspond to one or more cells. The tables may be stored in a

- 5 memory device such as a static random access memory (SRAM) device, another RAM device, a cache or any other desired memory device.

The swap operation 200 may be implemented in hardware or in software executed by a scheduler or other device. The software may furthermore be stored in an article of manufacture including a computer readable medium which, when executed,  
10 causes a processor to derive index values in accordance with the swap operation 200 as described herein. The software may also include instructions for populating the slots in the table.

The table on which the swap operation 200 is performed may be a port shaping table 250, such as shown in Figure 3, that maintains a schedule of traffic from various  
15 ports depending on their configured line rates. Each port may have a specific constant bandwidth requirement. An entry in a port shaping table is generally associated with a unit of bandwidth that is associated with a particular port. The port shaping table would have N maximum entries that it processes cyclically. For example if  $N = 2048$  the entries could be processed in order from 1 through 2048, and then the process could  
20 loop back to the first entry and start over where the step Z between all units of bandwidth from a particular port are the same.

The swap operation 200 of Figure 2 illustrates the use of the swap operation 200 in populating a configuration table. At 202 a list of sub-streams is prepared for insertion into the configuration table. The sub-streams may be sorted in decreasing order by  
25 size, with the sub-stream with the largest number of components going first to efficiently populate the configuration table. At 204, L represents the number of entries already inserted into the configuration table, having a total number of slots N, where N equals two to an x power and L is assigned the value zero before any entries are inserted. At 206 the first sub-stream for insertion is retrieved. At 208 the lower x bits of the binary  
30 representation of L are swapped to obtain L'. At 210 the number of components, K, of the sub-stream is obtained. K should also be expressible as a power of two where K equals two to the y power and K may be adjusted upwardly to arrive at such a value. At

5     212 the step for the sub-stream Z is calculated by dividing N by K and a counter w is initialized to zero. It should be noted that Z is also expressible as two to the x-y power.

At 214 to 218, an indexing loop may be repeated for each component of the sub-stream so that every component has an associated index for an entry into a slot in the table. Thus, at 214 an index value is determined by adding L' to w times Z. At 216 w is  
10     incremented by 1 and at 218 if w is less than K then the indexing at 214 and incrementing at 216 are repeated for the new value of w.

At 214 the components may also be identified in accordance with their respective index values. For example, an identifier may be entered into the table at the proper slot in accordance with the index value to serve as a pointer to the associated sub-stream  
15     component, such as an ATM cell or a unit of bandwidth from a particular port of on an ATM switch. Once all components are indexed for a sub-stream, a check is made at 220 to determine whether another sub-stream is available for processing. If another sub-stream is available, then the new sub-stream is accessed at 222 and the swap operation 200 returns to 210 to assign slots to the components of that other sub-stream.  
20     If no additional sub-streams are available for processing, then the swap operation 200 may stop at 224. It should be noted that the swap operation 200 may be repeated as desired to process all sub-streams.

Figure 3 shows a port shaper table 250 that has been populated in accordance with the swap operation. 28 ports have been allocated slots of bandwidth in the table  
25     250. Each entry in the table corresponds to a unit of bandwidth and includes the port number to which it is allocated. In the port shaper table 250, port number 1 receives four slots, or 4/32 of the total available bandwidth of the port shaper table 250. Port number 8 receives two slots, or 2/32 of the total bandwidth. Each of the other ports receives one slot or 1/32 of the bandwidth. All ports have a constant bandwidth  
30     requirement associated with them, and no port occupies a slot designated for another port. In this case the constant bandwidth requirement of port number 1 is twice as much as that of port 8 and four times greater than the other ports.

5           Figure 4 shows an embodiment of an ATM switch 300 in which an embodiment of the swap operation may be used to populate a table 302 in a scheduler 304. The ATM switch illustrated includes 28 ports 306 through which streams may be received from nodes coupled to those ports 306 and multiplexed through a multiplexer 312. Those ports 306 each have an associated constant bandwidth. A processor 308 will  
10 allocate slots in the table 302 to each of the 28 ports in accordance with the swap operation described in connection with Figures 1 and 2. Once the table 302 is populated it may, for example, look like the table 250 in Figure 3. The slots of the table 302 may be populated with identifiers that point to corresponding entries in a cell transmit table 310. The processor may then cycle through the table 302 and signal the  
15 cell transmit table 310 to transmit cells in the order of the entries in the table 302. The timing requirements of cell transmission are met by virtue of the placement of the entries into the table 302 in accordance with the swap operation while assuring that entries are made in the table without having to verify slot availability. In the example illustrated in Figure 3, the table 302 may be a port shaper table.

20           The processor 308 may execute the program instructions and process the data stored in memory coupled to the processor. In one embodiment, the instructions are stored in memory in a compressed and/or encrypted format. As used herein the phrase, "executed by a processor" is intended to encompass instructions stored in a compressed and/or encrypted format, as well as instructions that may be compiled or  
25 installed by an installer before being executed by the processor 308.

          Figure 5 shows an embodiment of a network 350 in which an embodiment of the swap operation may be employed. The network 350 includes a first ATM switch 352 coupled to a LAN (local area network) switch 354 and a second ATM switch 356 by, for example, ports such as the ports 306 illustrated in the ATM switch 300 of Figure 4. The  
30 first ATM switch 352 may, for example, be coupled to the second ATM switch 356 by way of an ATM backbone and may receive cells from the LAN switch 354 and transmit those cells to the second ATM switch 356. Thus, the first ATM switch 352 may perform the swap described herein to prepare a table for a cell scheduler to transmit cells

5 received from the LAN switch 354 to the second ATM switch 356. Alternately, the first ATM switch 352 may receive cells from the second ATM switch 356 and transmit those cells to the LAN switch 354 in accordance with the swapping operation and table preparation described herein.

10 The second ATM switch 356 is also coupled to a workgroup ATM switch 358 and may operate on cells received from or transmitted to the workgroup ATM switch 358 utilizing the swapping operation and table preparation described herein. Two workstations 360 and 362 are also coupled to the workgroup ATM switch 358 such that cells transmitted from either of those workstations 360 and 362 to the workgroup ATM switch 358 or received by either of the workstations 360 and 362 from the workgroup  
15 ATM switch 358 may be processed in the workgroup ATM switch 358 in accordance with the cell swapping and table preparation described herein.

Thus, ATM switches with schedulers may populate schedule tables in accordance with the swapping operation and table preparation described herein. In doing so, the tables may dictate the timing of cell transmission. End users may thus  
20 send real-time data with constant bandwidth requirements such as voice or video data through ATM switches or other devices utilizing the cell swapping and table preparation described herein by way of ports in those devices configured for various constant bandwidth transmission rates. The bandwidth and timing requirements of the ports and ultimately of the end user data are furthermore met through the application of the swap  
25 operation to the population of tables in the schedulers.

While the systems, apparatuses, and methods of preparing a table for a cell scheduler have been described in detail and with reference to specific embodiments thereof, it will be apparent to one skilled in the art that various changes and modifications can be made therein without departing from the spirit and scope thereof.  
30 Thus, it is intended that the modifications and variations be covered provided they come within the scope of the appended claims and their equivalents.